Accessibility

Introduction

In the Play+ ecosystem, accessibility is not an afterthought—it's a core component of our "Design that Breathes" philosophy. We believe that great design is inclusive design, and an application that isn't accessible to everyone is incomplete. This guide is based on the principle of Inclusive by Design.

The playa11y helper provides a foundational toolkit to automate accessibility best practices, reduce boilerplate code, and empower developers to build experiences that are usable by everyone, regardless of their abilities. This aligns directly with our Inclusive pillar by embedding WCAG 2.2 AA standards directly into the development workflow, ensuring every user feels a sense of belonging and trust.

The Three Pillars of Accessibility Enforcement

1. Automated Tooling (Linter & CI)

- Our playlint ESLint config includes the jsx-a11y plugin.
- All PRs run an axe-core audit.
- If critical violations are found, the build fails (non-negotiable quality gate).

2. Accessible-by-Default Components

- All core Play+ components: Include full keyboard navigation Clear focus states Compliant contrast ratios Correct ARIA roles (e.g., modals, tabs)

3. Developer Practices & the playa11y Toolkit

- Developers own the final layer of a11y.
- This guide and toolkit make it easy to implement best practices.

Package Info

Package / Path	Description
Golden Path (Recommended)	Pre-installed: /system/play.a11y.ts
Uplift Path (Coming Soon)	@playplus/a11y

Folder Reference

Directory	Purpose & Guidelines
/system/	Core Play+ helpers (e.g., play.a11y.ts)
/config/	User-overridable configs (e.g., play.a11y.config.json)

Helper - Pillars Alignment

Pillar	How This Helper Aligns
Inclusive	Primary Pillar: Enforces WCAG 2.2 AA for diverse ability support
Intuitive	Ensures predictable, seamless experience for assistive technologies
Engaging	Allows graceful degradation of motion, ensuring delight without discomfort

Helper Overview

The playa11y helper is a lightweight utility that abstracts the complexity of accessibility. It automates common tasks, avoids boilerplate, and prevents errors.

Key Features

- Live region announcements
- Safe focus management
- Reduced motion detection
- Dev-only contrast ratio checks

Config Options

Config Variable	Default Value	Description	Recommended Value
liveRegionPoliteness	"polite"	Default level for screen reader messages	"polite"
focusRing.color	"#E91E63"	Color of focus ring (from token)	$color-border-focus
focusRing.offset	"2px"	Distance between element and focus ring	$space-1
focusRing.width	"2px"	Thickness of focus ring	2px

Helper Methods

Core Methods

Method Name	Description	Signature
announce	Announces a message to screen readers	announce(message: string, politeness?: 'polite' | 'assertive'): void
focus	Safely moves focus to an element	focus(element: HTMLElement | null | undefined): void
trapFocus	Traps focus within a container	trapFocus(container: HTMLElement): () => void
restoreFocus	Restores previous focus	restoreFocus(): void
prefersReducedMotion	Detects user's motion preference	prefersReducedMotion(): boolean
validateContrast	Validates color contrast ratios	validateContrast(color1: string, color2: string, context: string): boolean

Method Name	Description	Signature
getContrast	Calculates contrast ratio (dev only)	getContrast(color1: string, color2: string): number

Angular Integration

PlayA11yService

Angular service wrapper that integrates with Play+ logging and provides component-specific accessibility utilities.

Accessibility Directive

Automatically validates accessibility and provides utilities.

Accessibility Pipe

Provides accessibility utilities in templates.

Usage Examples

React: Managing Focus and Announcements

Angular: Using Angular CDK

Basic Usage Examples

Additional Info

Why We Created This Helper

Accessibility is critical for inclusive design, but implementing it correctly can be complex and error-prone. Without a dedicated system, developers would need to:

- Manually implement focus management for every interactive component
- Write boilerplate code for screen reader announcements
- Remember to check motion preferences in every animation
- Manually validate color contrast ratios

This leads to inconsistent implementations and accessibility gaps. The playa11y helper automates these common patterns and provides a simple, consistent API that makes accessibility the default path.

WCAG 2.2 AA Compliance

All Play+ applications must meet WCAG 2.2 AA standards, which include:

- Perceivable : Information must be presentable to users in ways they can perceive
- Operable : User interface components and navigation must be operable
- Understandable : Information and operation of user interface must be understandable
- Robust : Content must be robust enough to be interpreted by a wide variety of user agents

Best Practices

Semantic HTML Structure

Proper Heading Hierarchy

Semantic Elements

Forms and Inputs

Proper Labels

Error Handling

Keyboard Navigation

Focus Management

Skip Links

Images and Media

Alt Text

Video and Audio

Dynamic Content

Live Regions

Focus Management for Dynamic Content

Forbidden Patterns

Missing Alt Text

Improper Heading Structure

Non-semantic Elements

Missing Labels

Keyboard Traps

Required Patterns

Semantic HTML

Proper Labels

Focus Management

Live Announcements

Reduced Motion Support

Testing Accessibility

Automated Testing

Manual Testing Checklist

Keyboard Navigation

- All interactive elements are keyboard accessible
- Focus order matches visual order
- No keyboard traps
- Skip links work properly

Screen Reader Testing

- Use VoiceOver (macOS) or NVDA (Windows)
- Navigate with keyboard only
- Check that all content is announced
- Verify form labels and error messages

Visual Testing

- Test at 200% zoom
- Check color contrast ratios
- Verify focus indicators are visible
- Test with high contrast mode

Monitoring and Analytics

Accessibility Metrics

- WCAG Compliance : Percentage of components meeting WCAG 2.2 AA
- Keyboard Navigation : Success rate of keyboard-only navigation
- Screen Reader Compatibility : Issues reported by screen readers
- Color Contrast : Percentage of text meeting contrast requirements

Reporting

Accessibility issues are automatically logged and can be sent to:

- Play+ logging system
- Accessibility monitoring dashboards
- CI/CD pipelines
- Development team notifications